Microsoft
.net developer
training

Creating Windows® Applications

Agenda: Day 2

Module 01: What is Microsoft® .NET?

Module 02: Visual Studio® .NET

Module 03: The .NET Languages

Module 04: Building ASP® .NET Applications

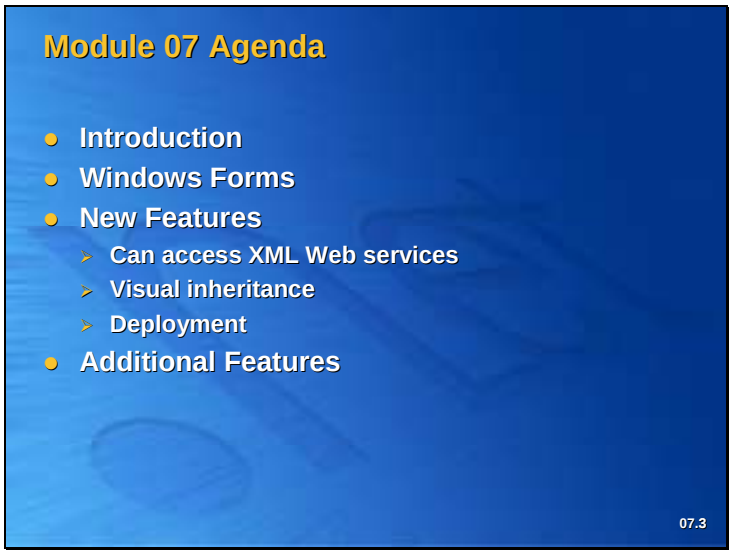
Module 05: Using Data Access in .NET Applications

Module 06: Object Oriented Concepts in .NET

▶ Module 07: Creating Windows Applications

Module 08: Using XML in .NET

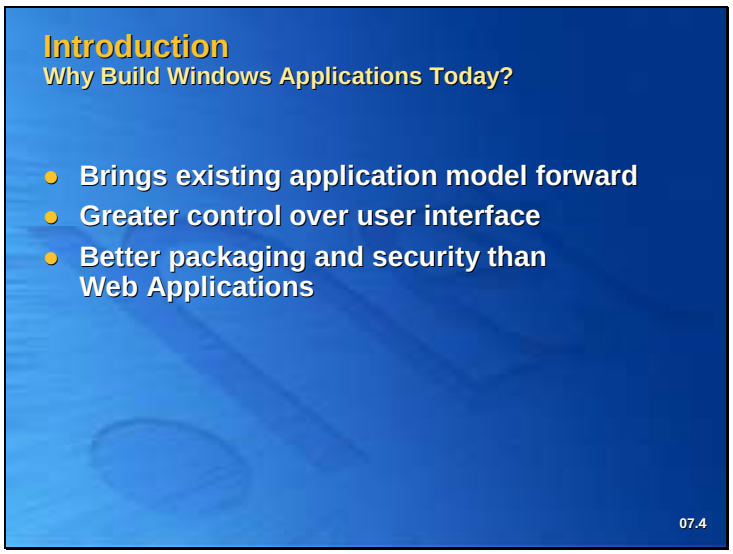
 ©2002 Microsoft Corporation



Module 07 Agenda

What you will learn:

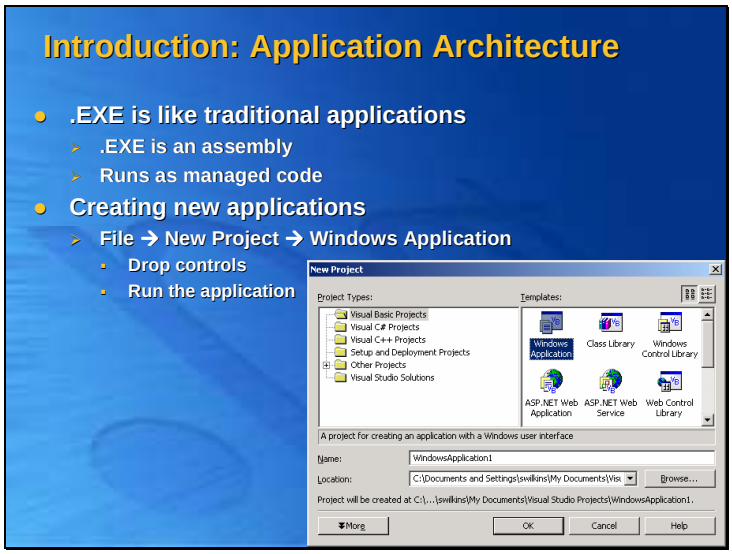
- Create Windows Applications in Visual Studio® .NET
- Create forms using the Windows Forms Designer
- Access XML Web services from you Windows Application
- Deploy your Windows Application


Introduction
Why Build Windows Applications Today?
Brings existing application model forward
Greater control over user interface
Better packaging and security than Web Applications
07.4

Introduction

- Why Build Windows Applications Today?

 ©2002 Microsoft Corporation

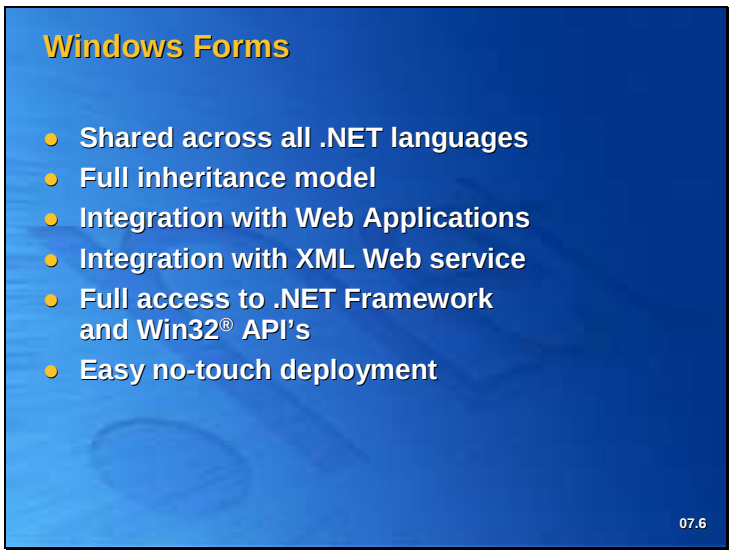


Introduction

Application Architecture

It remains possible to create Windows Applications in .NET. For example, you can use the .NET Framework to develop the following types of applications and services:

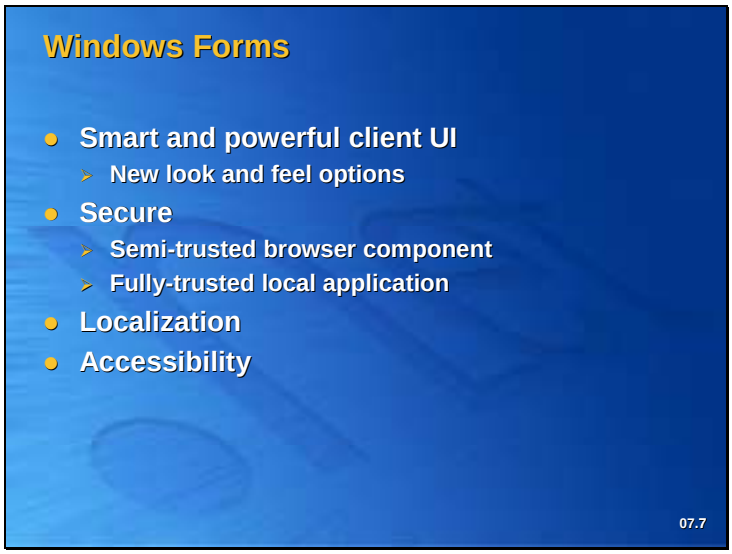
- Console applications
- Scripted or hosted applications
- Windows GUI applications (Windows Forms)
- ASP.NET® applications
- Web services
- Windows 2000 and Windows NT services



Windows Forms

Windows Forms leverage various new technologies including a Common Application Framework, Managed Execution Environment, Integrated Security and Object-oriented Design Principles. Developers can create Windows Forms applications much like today's Visual Basic-based forms, although with Windows Forms, they get the same level of productivity in **all of the languages** in Visual Studio®, not just Visual Basic. Windows Forms offers full support for quickly and easily connecting to **XML Web services** and building rich, data-aware applications based on the ADO.NET data model.

 ©2002 Microsoft Corporation



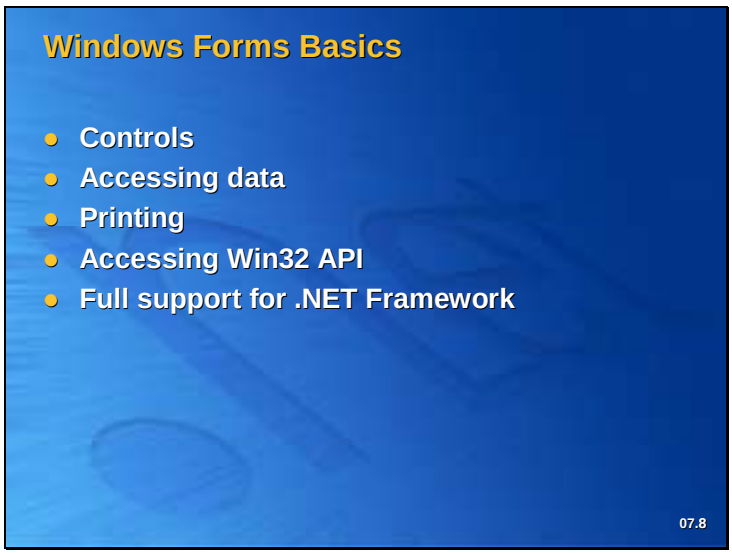
Windows Forms

Unlike ActiveX controls, Windows Forms control enjoys semi-trusted access to a user's machine so binary or natively executing code can access, but not undermine some of the resources on the user's system such as GUI elements and limited file access.

You can deploy your application's resources in satellite assemblies, which only contain resource files. In the satellite assembly deployment model, you create an application with one default assembly (the main assembly) and several satellite assemblies. You should create one satellite assembly for each culture that the application supports. Because the satellite assemblies are not part of the main assembly, you can replace or update resources corresponding to a specific culture without replacing the application's main assembly.

Carefully determine which resources will make up your application's default resource assembly. Because it is a part of the main assembly, any changes to it will require you to replace the main assembly. If you fail to provide a default resource, an exception will be thrown when the resource fallback process attempts to find it. In a well- designed application, using resources will never throw an exception.

Windows Forms implement Microsoft Active Accessibility® programming interfaces and it is now simpler to build applications that support accessibility aides such as screen readers.



Windows Forms Basics

Windows Forms provides the following features:

- **A rich set of controls**

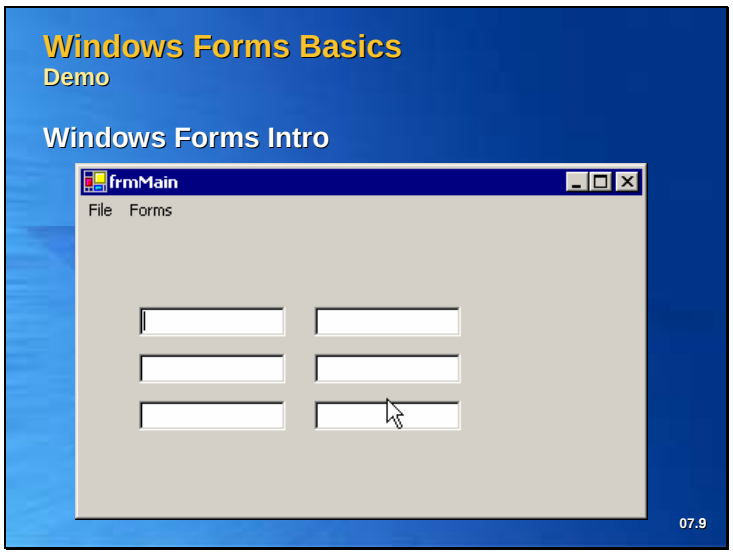
  Standard controls: Label, LinkLabel, TextBox, RadioButton, Checkbox, GroupBox, Panel, ListBox, CheckedListBox, ComboBox, Button, Menu, Panel PictureBox, MaskedEdit, Up-Down, Scrollbar, Splitter, Tooltip, Menus, Help and PropertyGrid

- **Common dialogs and RichEdit**

  Common Controls: Progress Bar, Status Bar, Toolbar, ListView, TreeView,Tab Control, DateTime Picker, Calendar, Tooltip, Trackbar and ImageList

- **Data controls:** ListBox, ComboBox, DataGrid, ErrorProvider
- **Advanced graphics support** – GDI+
- **"Money 2000" Flat look**
- **Advanced layout**
- **Visual forms inheritance**
- **Printing support**
- **Accessibility**
- Windows Forms also offers a printing framework that makes printing simple, including a Print Preview window with the Print Preview control
- Windows Forms are also extensible. You can extend and enhance the controls and framework, and easily add design-time behavior.

 ©2002 Microsoft Corporation



Windows Forms Basics

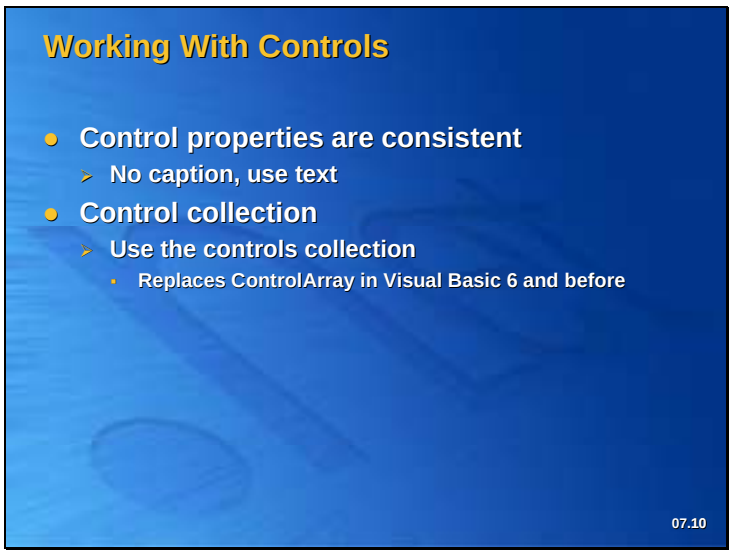
Demo

Windows Forms Intro

Building a simple Windows Forms Application

The purpose of this demo is to illustrate:

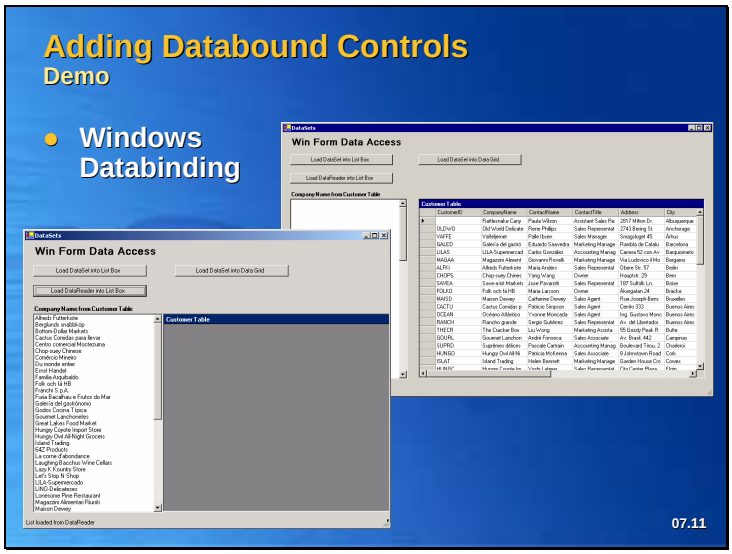
- Building a Windows Forms Application
- How the process is similar to a V 6 application
- How to use simple controls
- How to show forms
- Show the window, always staying on top, and show a transparent background in a window (Set different values for the Opacity property of the Form to show how it affects the visibility of a form.)



Working With Controls

New controls including Link Label, Tray Icon and Print Preview provide additional common functionality for developers. **Link Label** provides linking similar to HTML to a specified URL. Text displayed using this control will appear underlined and the cursor will change to a hand as the mouse moves over it, firing an actionable event when the text is clicked. **Tray Icon** enables developers to create applications that run in the Windows tray similar to the Microsoft SQL Server™ Service Manager.

 ©2002 Microsoft Corporation



Adding Databound Controls

Demo

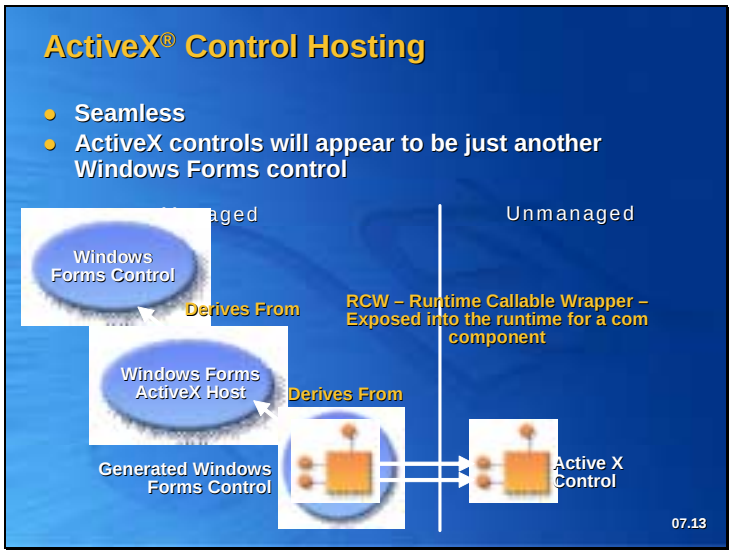
- Working with Bound Controls
- Data binding using components

The purpose of this demo is to illustrate:

- How to data bind to a windows form control
- How components can be used with data binding
- Flexibility of the data grid

New Features of Windows Forms Topics

 ©2002 Microsoft Corporation

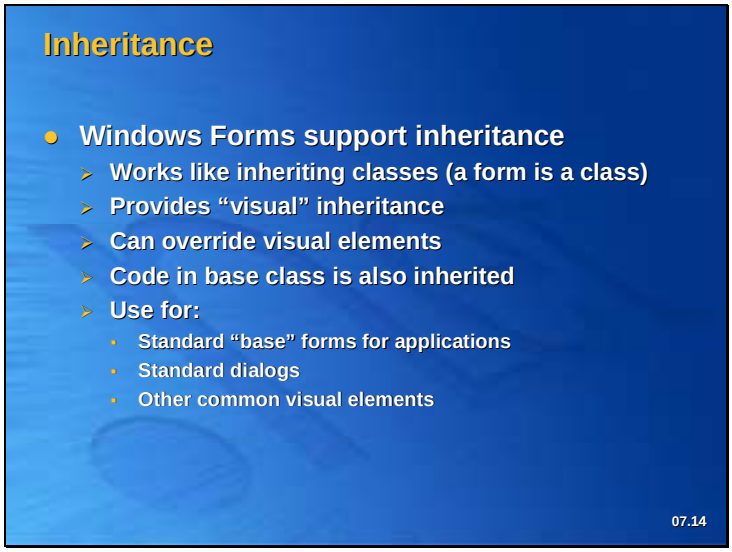

ActiveX® Control Hosting
Seamless
ActiveX controls will appear to be just another Windows Forms control
Unmanaged
Windows Forms Control
Derives From
RCW – Runtime Callable Wrapper – Exposed into the runtime for a com component
Windows Forms ActiveX Host
Derives From
Generated Windows Forms Control
Active X Control
07.13

ActiveX Controls on Windows Forms

While Windows Forms have been optimized to host Windows Forms controls, you still are able to use ActiveX controls on Windows Forms. Some of the things to keep in mind when planning an application that uses ActiveX controls are listed below:

Security – The Common Language Runtime has been maximized with regard to Code Access Security. Applications featuring Windows Forms can run in a fully trusted environment without issue and in a semi-trusted environment with some of the functionality curtailed. However, ActiveX controls on Windows Forms cannot take advantage of these security enhancements. Running an ActiveX control requires unmanaged code permission. For more information about Security and unmanaged code permission, see SecurityPermissionAttribute class.

Total Cost of Ownership (TCO) – Any ActiveX controls added to a Windows Form are deployed with the Windows Form, and this can increase the size of the file(s) created. Additionally, the use of ActiveX controls on Windows Forms requires writing to the registry, which is more invasive to a user's computer than the Windows Forms controls.

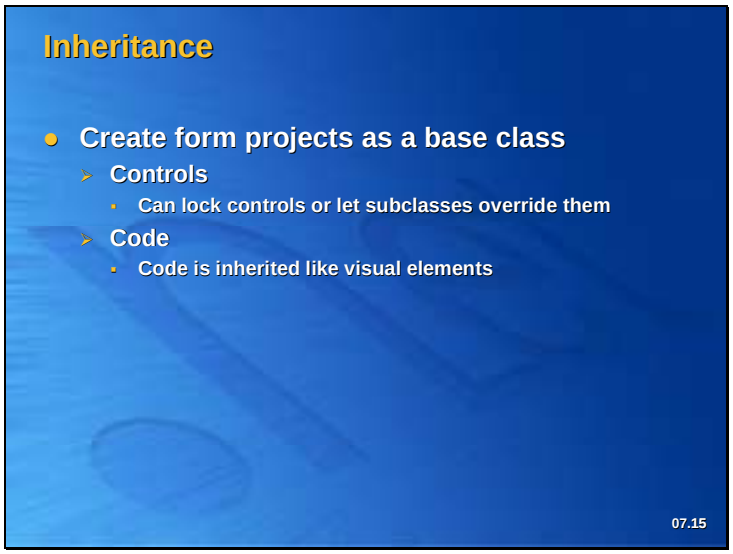


Inheritance

On some occasions, you may decide that a project calls for a form similar to one that you have created in a previous project, or you may want to create a basic form with settings such as a watermark or certain control layout that you reuse with each iteration containing modifications to the original form template. Form inheritance enables you to create a base form and inherit from it and make modifications to it while preserving the necessary original settings.

For example, an organization could define a standard base form that contains items such as the corporate logo and perhaps a common toolbar. This form can be used by developers through inheritance and extended to meet the requirements of specific applications while promoting a Common User Interface across the organization. The creator of the base form can specify which elements can be extended and which must be used as is, ensuring that the form is reused appropriately.

 ©2002 Microsoft Corporation



Inheritance

Creating new Windows Forms by inheriting from base forms is a useful way to duplicate your best efforts without going through the process of entirely recreating a form.

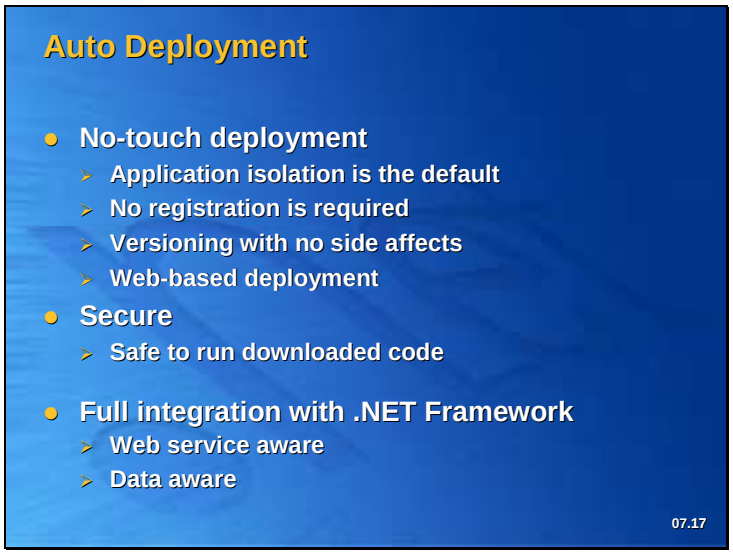
To inherit a form programmatically:

- In your class, add a reference to the namespace containing the form you wish to inherit from
- In the class definition, add a reference to the form from which to inherit. The reference should include the namespace that contains the form, followed by a period and then the name of the base form itself

Visual Inheritance

Demo

 ©2002 Microsoft Corporation



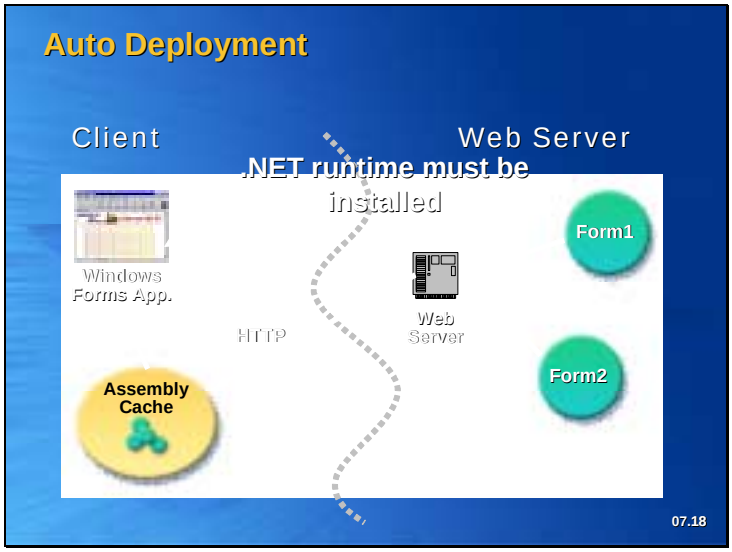
Auto Deployment

Code Access Security makes Windows Forms controls safe

Execution is controlled by:

- Permissions
- Permission sets
- Code groups (Zones)
- Code groups have permission sets

See Module 11: Security in .NET for more information on Code Access and Role-based Security.



Auto Deployment

With a Windows Forms application, it is unnecessary to deploy an application to the end user's desktop. Instead, a user can invoke the application simply by typing a URL into a browser. The application will download to the client machine, run in a secure execution environment and can remove itself upon completion.

Auto deployment needs no installation:

- Store the assembly on the Web server
- In the applications, use Assembly.LoadFrom
- The application is downloaded, form by form as needed via the HTTP Web server
- It is stored in the GAC and the Internet download cache
- It is only downloaded as needed or when updated
- Dependent components are automatically downloaded to the client so the application “trickles” onto the client

If you use an installer, the installer can download the .NET runtime and install it. It’s just a bit large.

 ©2002 Microsoft Corporation

Summary
You still can build Windows Applications
Smart User Interface
Deployment
Easy to install
Solves DLL Hell
Code Access Security makes downloadable code safe
Integration with XML Web services
07.19

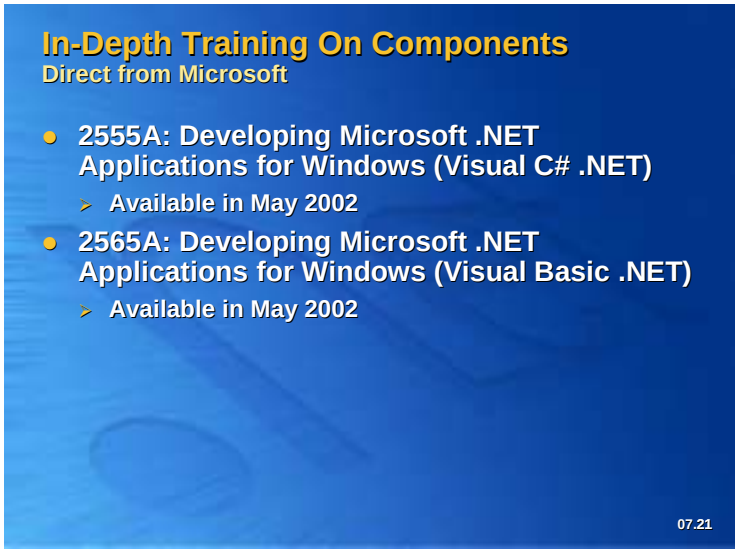
Summary

Resources

An introduction to Windows Forms - Creating basic forms in a Windows Application
http://msdn.microsoft.com/vstudio/nextgen/technology/winforms.asp

The Basics of Windows Forms - http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vbcon/html/vbconintroductiontowfcforms.asp

 ©2002 Microsoft Corporation


In-Depth Training On Components
Direct from Microsoft
2555A: Developing Microsoft .NET Applications for Windows (Visual C# .NET)
Available in May 2002
2565A: Developing Microsoft .NET Applications for Windows (Visual Basic .NET)
Available in May 2002
07.21

In-Depth Training On Components

Direct from Microsoft

 ©2002 Microsoft Corporation